PL-TN--92-1005

PL-TN--
92-1005

(2)

STABILIZED IODINE FLOW FOR LONG RUN TIME CHEMICAL OXYGEN-IODINE LASERS

M. P. Murdough
C. A. Helms

October 1992

Final Report

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

PHILLIPS LABORATORY
Lasers and Imaging Directorate
AIR FORCE MATERIEL COMMAND
KIRTLAND AIR FORCE BASE, NM 87117-6008

PL-TN--92-1005

This final report was prepared by the Phillips Laboratory, Kirtland Air Force Base, New Mexico, under Job Order 3647LA02. The Laboratory Project Officer-in-Charge was Mr Charles A. Helms (LIDB).

This report has been reviewed and is approved for publication.

Charles A. Helms

CHARLES A. HELMS
Project Officer

FOR THE COMMANDER

Steven E. Lamberson

STEVEN E. LAMBERSON
Lt Colonel, USAF
Chief, Applied Laser Technology

Charles B. Hogge

CHARLES B. HOGGE, DV6
Acting Director, Lasers and Imaging
Directorate Branch

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
October 1992

3. REPORT TYPE AND DATES COVERED
Final 1 Oct 90 - 31 Jan 92

4. TITLE AND SUBTITLE
STABILIZED IODINE FLOW FOR LONG RUN TIME CHEMICAL OXYGEN-IODINE LASERS

5. FUNDING NUMBERS
PE: 63605F
PR: 3647
TA: LA
WU: 02

6. AUTHOR(S)
M. P. Murdough and C. A. Helms

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Phillips Laboratory
Kirtland Air Force Base, NM 87117-6008

8. PERFORMING ORGANIZATION REPORT NUMBER
PL-TN--92-1005

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A computer-controlled iodine flow rate system has been built for the chemical oxygen-iodine laser. An optical measurement was used to determine iodine flow rate and helium flow rate was varied to maintain a constant iodine flow rate during long laser tests. The automated system worked well over a wide range of iodine flow rates and initial conditions.

14. SUBJECT TERMS
Chemical Oxygen-Iodine Laser, Iodine Flow Measurement

15. NUMBER OF PAGES
28

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	SAR

CONTENTS

FIGURES

1.0 INTRODUCTION

The Phillips Laboratory Chemical Laser Facility, Kirtland AFB, New Mexico, has been engaged in the development of chemical-oxygen iodine lasers (COIL) for over 10 years. In the past, these lasers have required iodine flow rates of 0.3 to 25 mmole/s for lasing times up to 10 s. Recent developments in COIL technology now require the delivery of stable iodine flows for much longer times. The Chemical Laser Facility has developed a feedback-control system which maintains stable iodine flow rates for long times despite fluctuations in iodine temperature and liquid level.

2.0 IODINE PRODUCTION AND MEASUREMENT

2.1 GASEOUS IODINE PRODUCTION

The COIL devices operated at the Chemical Laser Facility require iodine flow rates of 0.3 to 25 mmole/s. The iodine required for these lasers is generated by bubbling helium diluent through a container filled with liquid iodine. Semiconductor grade iodine is heated in a Hastelloy vessel (boiler) to temperatures from 115 to 145°C, producing liquid iodine. Note that the melting point of iodine is 113.5°C and the boiling point is 184°C. A regulated flow of preheated helium is passed through the boiler to entrain the iodine and transport it to the device. The flow rate of iodine delivered is determined by the temperature of the liquid iodine and the amount of helium flowing through the boiler. These two variables (temperature and helium flow rate) can easily be controlled to set the iodine production from the boiler.

Figure 1 shows the original arrangement of the iodine boiler, helium diluent systems, and diagnostic equipment used to measure iodine flow rates. Two valves are opened simultaneously to allow helium to enter the boiler through the sparger and exit at the top. The flow rate of helium through the boiler was controlled by moving the helium through a supersonic orifice at a known pressure. Knowing the cross sectional area of the orifice, the pressure upstream of the orifice, the molecular weight of the gas, and the temperature

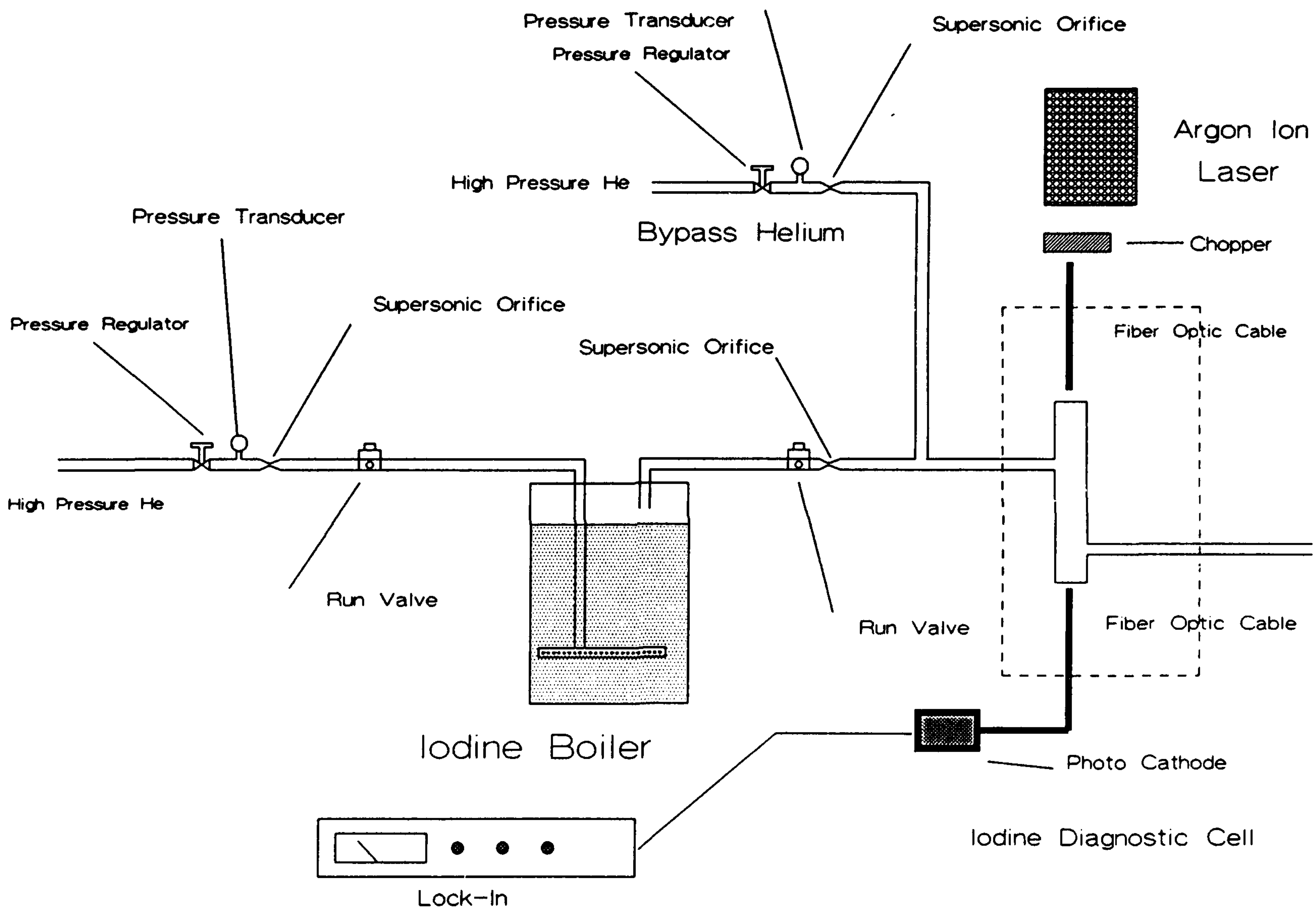

Figure 1. Schematic illustration of original iodine system.

of the gas, it was possible to calculate the flow rate of helium via the sonic orifice equation.

In COIL lasers, a constant helium flow through the iodine system is necessary in order to control the penetration of the iodine/helium jets into the main gas flow. This requires an additional helium "bypass" flow. If the helium flow through the boiler is changed, the bypass flow must compensate to keep the total flow constant. A supersonic orifice mounted between the boiler and the junction where the bypass was added prevented the bypass flow from affecting iodine production. This bypass helium does, however, affect the concentration of iodine at the diagnostic cell where flow rate was measured.

2.2 REAL-TIME IODINE FLOW RATE MEASUREMENT

Determination of the iodine flow rate required the use of a diagnostic cell mounted downstream from the iodine boiler (Fig. 2). All of the iodine system flow, including the bypass helium, flowed through the diagnostic cell. The molecular iodine concentration was measured using a Beer's law absorption performed at a wavelength of 488 nm. The fraction of light absorbed is related to the number density of iodine present in the cell by:

$$I = I_0 e^{-\sigma \ell n} \tag{1}$$

where σ is the absorption cross section, l is the path length in centimeters, and n is the number density per cubic centimeter. Using the ideal gas law and the cell temperature, the iodine number density can be converted to partial pressure (Eq. 2). The iodine partial pressure, total cell pressure, and helium flow rate can then be used to solve for the iodine molar flow rate (Eq. 3). An argon ion laser at 488 nm was used as the light source.

$$P_{I_2} = (n/V)RT \tag{2}$$

$$\dot{n}_{I_2} = \dot{n}_{He}\left(\frac{P_{I_2}}{P_{cell}}\right) \tag{3}$$

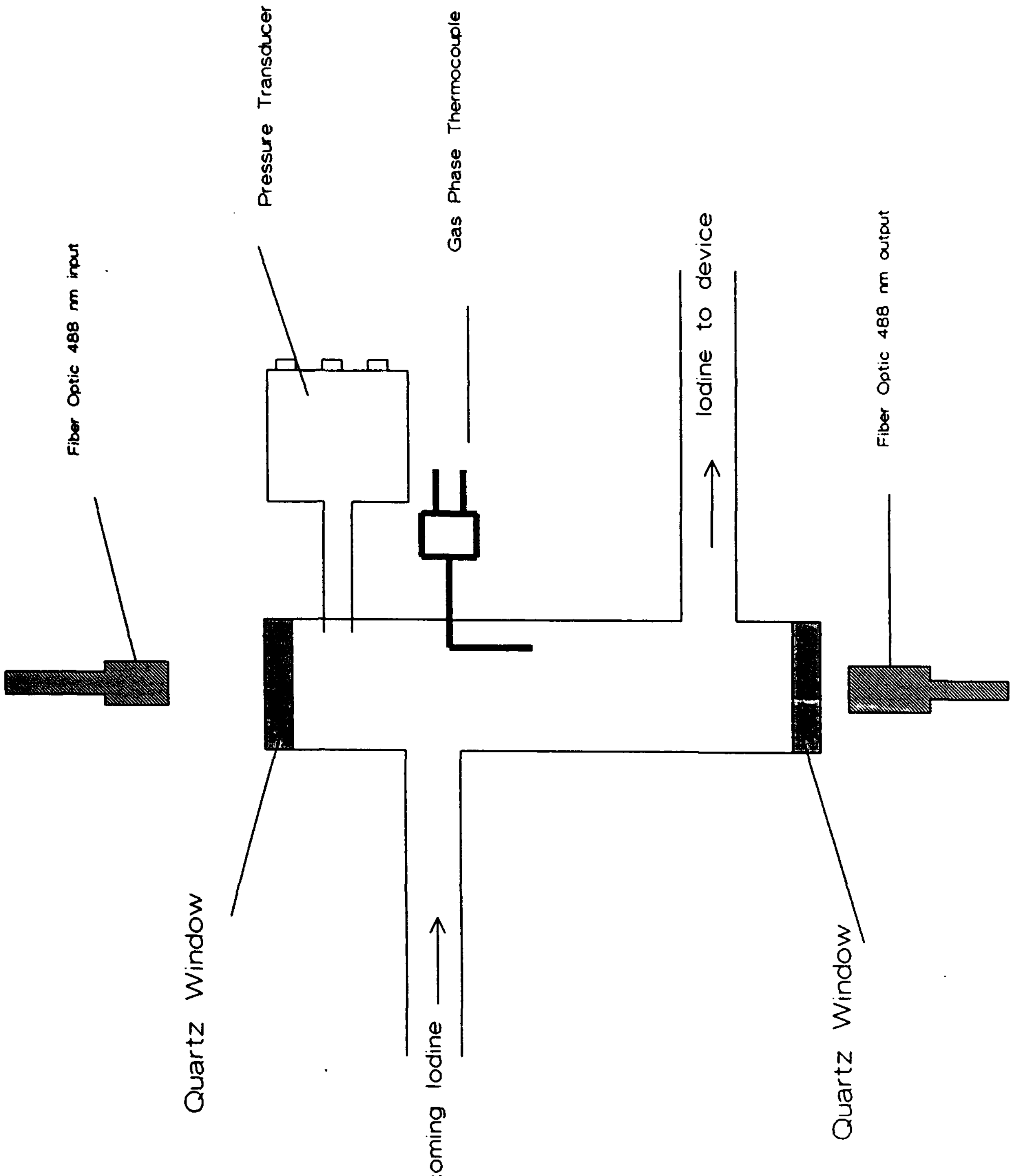

Figure 2. Details of iodine diagnostic cell.

3.0 IODINE CONTROL

The system which has been developed to control iodine flow depends on the same physical parameters as previous systems. However, a variety of new devices were introduced which allow for easier manipulation of helium flow rates. The signals required for real-time iodine measurement and control were sent to a computer via an analog-to-digital (A/D) board. A mass flow controller was used to replace the pressure transducer and sonic orifice for helium flowing through the boiler. For small lasers, a second mass flow controller replaced the sonic orifice in the bypass helium flow. Mass flow controllers did not have the capacity to meter the helium required for the larger device (RADICL). This problem was solved by delivering roughly 80 percent of the bypass helium using the standard orifice technique and controlling the remaining 20 percent with the flow controller. The orifice helium flow only needed to be considered when calculating total helium flow through the diagnostic cell.

3.1 DIAGNOSTICS AND DATA ACQUISITION

In order to control iodine flows in real-time, it was necessary to monitor a variety of signals. Table 1 lists all the measurements collected by the computer.

Table 1. Signals monitored by computer.

Measurement	Instrument	Signal Level
Iodine Cell Pressure	Pressure Transducer	0-10 V
Iodine Cell Temperature	Thermocouple	V
Boiler Temperature	Thermocouple	V
488-nm Iodine Absorption	Lock-in Amplifier Voltage	0-10 V
Bypass Orifice Pressure	Pressure Transducer	0-10 V
Bypass Helium Mass Flow	Mass Flow Controller	0-5 V
Boiler Helium Mass Flow	Mass Flow Controller	0-5 V

In addition to recording the signals shown in Table 1, the computer controlled the mass flow controllers through two digital-to-analog (D/A) channels. Thermocouple output voltages were converted to temperatures using a polynomial fitting routine for T-type thermocouples. The signals for these various devices were connected to a DT707-T A/D board which was interfaced to a Data Translation DT2805 card in a 80286-based desktop computer. Data reduction and control were done using the ASYST programming language.

3.2 HELIUM MASS FLOW CONTROL

Mass flow controllers were used to replace the pressure transducer and sonic orifice concept of previous devices (Fig. 3). This allowed fast, accurate changes in helium flow rates. The calibrated mass flow controllers used were Unit Mfg. model UFC-3020, 100 standard liters per minute (74.4 mmole/s), calibrated for nitrogen. Manufacturer-supplied conversions allow the controllers to be used for helium. The full scale throughput for helium with these controllers was 107 mmole/s. The controllers were rated accurate to +/-1 percent of full scale. A Unit URS-100 Power Supply Readout was used to drive the mass flow controllers and supply feedback to the data acquisition computer.

3.3 PROGRAMMING

3.3.1 Theory

The method for controlling the iodine flow rate was relatively straight-forward. A logic flowchart is shown in Figure 4. The appendix contains a listing of the ASYST computer program. The underlying assumption made was that iodine flow at a given boiler temperature is linearly proportional to the flow of helium through the boiler. This approximation proved to be sufficient for the operating range. The computer compared the user-desired iodine flow rate with the current flow rate. Then it adjusted the helium flow rate through the boiler by commanding the mass flow controller to either increase or decrease helium flow proportionally with the change needed. A variable dampening factor was used to help prevent oscillations in flow rate. The dampening factor was nominally set at 0.8. For example, if the desired iodine

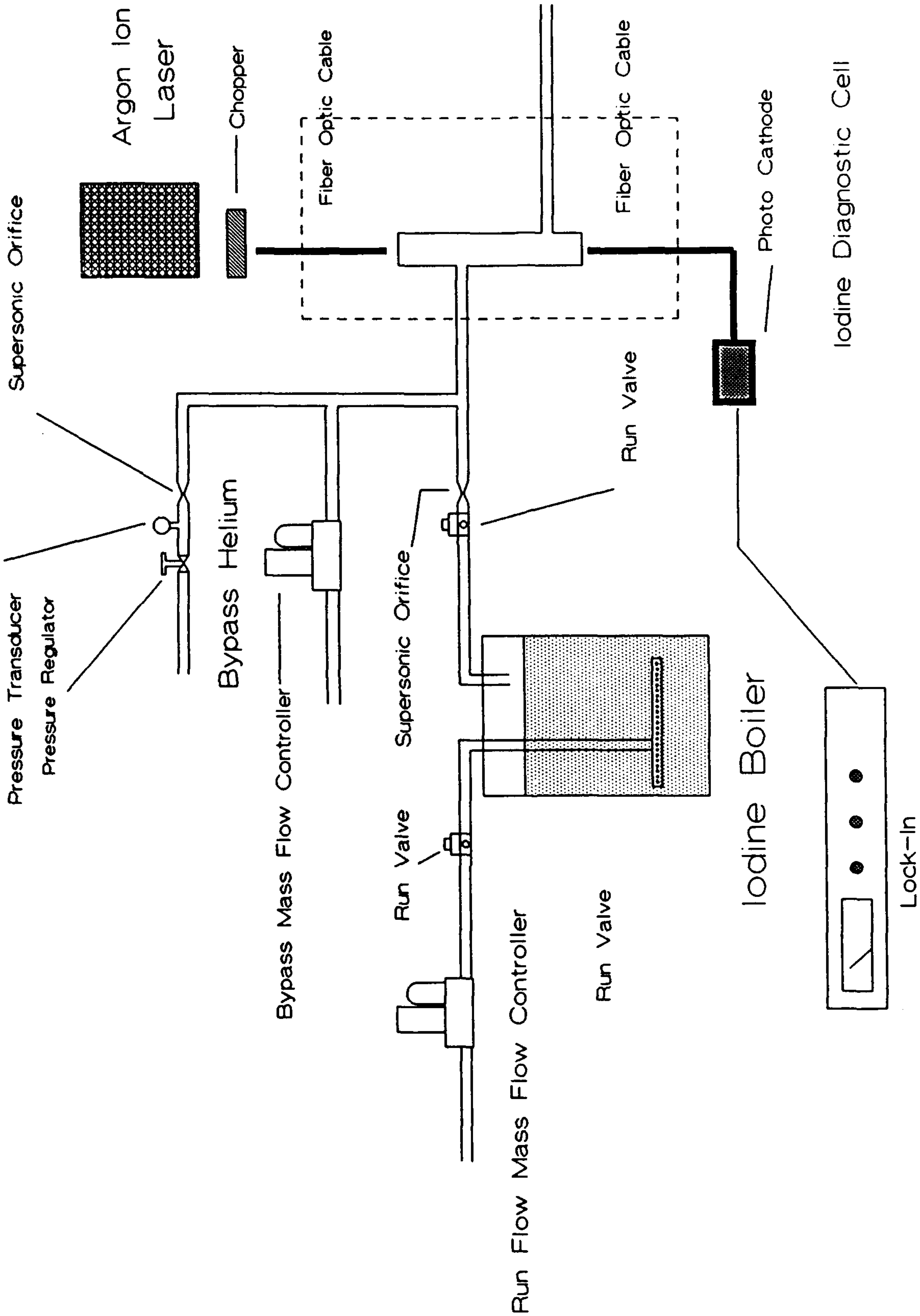

Figure 3. Schematic illustration of revised iodine system.

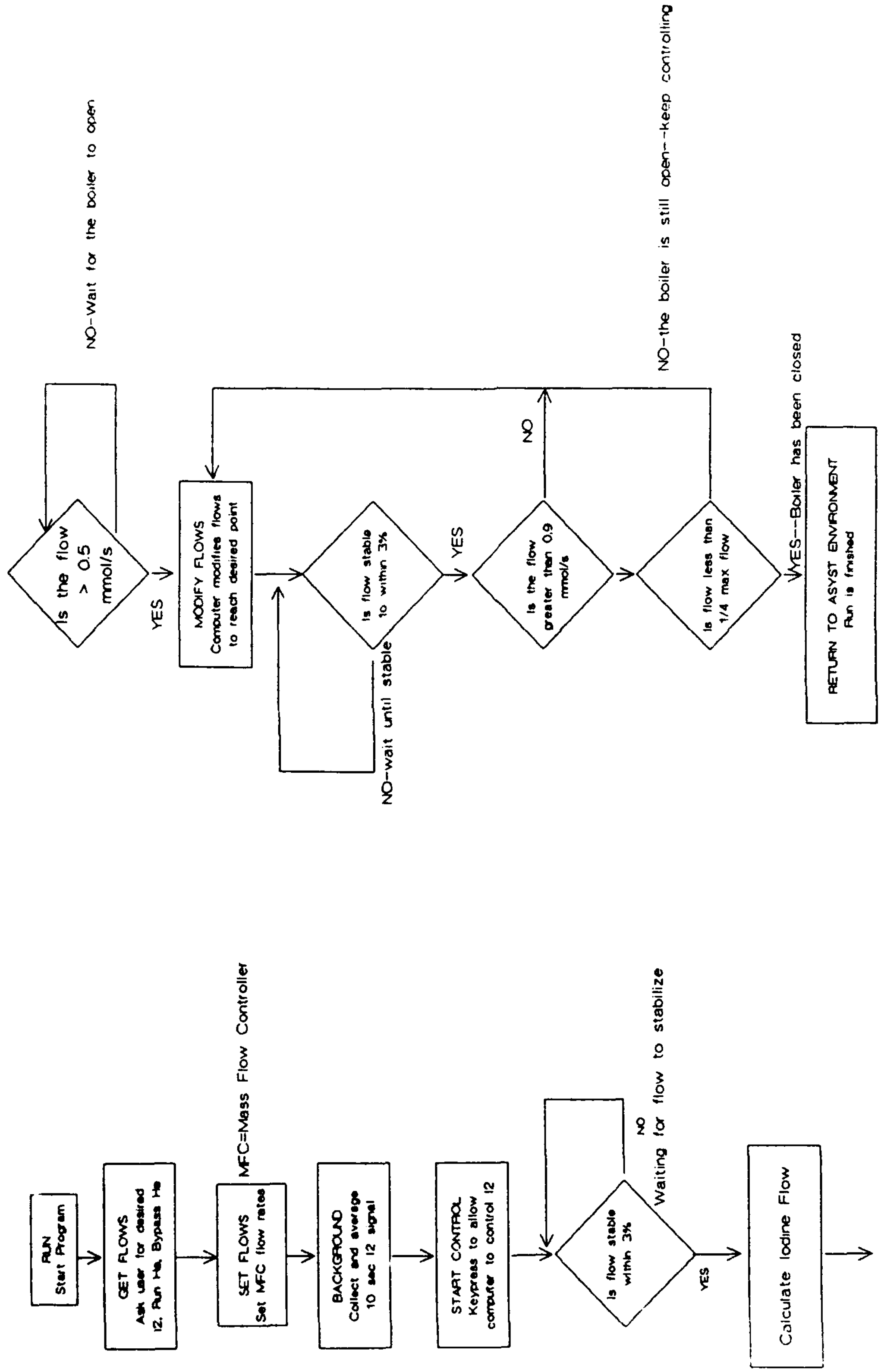

Figure 4. Flowchart of iodine control logic.

flow rate was 20 percent lower than the actual flow rate, the run helium flow would be decreased 16 percent and the bypass helium would be increased the same amount (not the same percentage). As the run progressed, the continuous feedback and adjustment compensated up for changes in boiler iodine production. In order to maintain a constant penetration as described previously, the total helium flow was held constant.

With the need for long runs, i.e., >20 s, it becomes necessary to adjust the helium flow rate during the run to maintain a constant iodine flow rate. This comes about for a variety of reasons. Between tests the boiler is closed by valves on either end. During this time the head space above the liquid reaches equilibrium iodine vapor pressure. When a run is initiated, the saturated gas in the head space is the first to be sent into the device. This effect generates a temporary overshoot in iodine flow. As the run progresses, the flow stabilizes at some evaporation rate determined by temperature and helium flow rate. As the run continues into the 40-s range and beyond, there is a significant cooling of the entire boiler thermal mass resulting in a lower evaporation rate. Consequently, it was necessary to gradually increase the helium flow rate to maintain the iodine production rate. In addition, it is very difficult, if not impossible, to reproduce exact conditions within the boiler for any two otherwise identical tests. Because of this, the control system must determine iodine flow rate in real-time to compensate for changing boiler conditions.

3.3.2 Practice

3.3.2.1 Set Initial Points. Three flow rates must be established at the start of a test. These are the desired iodine flow rate, the helium flow through the boiler, and the total helium flow through the iodine system. The program asked for both the run and bypass helium flows and calculated the total iodine flow, which was held constant for the entire run.

3.3.2.2 Background Signal. After setting the initial flow rates, the computer waits for a key press to collect the background iodine absorption signal. Once initiated, the computer records the iodine absorption signal voltage coming from the lock-in amplifier for a 10-s period at a 10-Hz collection rate. It then averages these values to determine a background iodine absorption signal (I_0 in Eq. 1). Once finished, it displays the average background signal and waits for another key press to begin controlling flows.

3.3.2.3 Calculating Iodine Flow Rates. After the key press, the computer continously calculates the iodine flow rate and displays it on the screen. For each calculation, it takes in the current data from the required diagnostics listed in Table 1 and uses Equations 1 through 3 to calculate iodine flow. The computer waits until the measured iodine flow rate is >1.0 mmole/s and has stabilized within 3 percent for a 3-s period before attempting to modify the iodine flow.

3.3.2.4 Modifying Flows. The modify flows routine works essentially as described in Subsection 3.3.1. The computer compares the measured iodine flow rate with the actual flow rate. It then increases or decreases the helium flow to the boiler by the ratio of the desired iodine flow to the real iodine flow with the resultant change in flow damped by a factor of 0.8. The computer remains in the modification mode and keeps the iodine flow rate at the desired set point. The computer interprets a rapid uncontrollable drop in iodine flow rate as the end of the test and stops attempting to control the flow rates.

4.0 RESULTS AND DISCUSSION

A typical example of uncontrolled iodine flow on the RADICL device is shown in Figure 5. The 30-percent drop in flow rate in 60 s is caused by the drop in iodine temperature due to the evaporation process in the boiler.

A similar iodine flow case on the radical device using feedback control is shown in Figure 6. The initial spike due to the saturated plug of iodine vapor described earlier is evident during the first few seconds. This flow of ~6 mmole/s could be maintained until the mass flow controller was unable to compensate for the drop in iodine temperature.

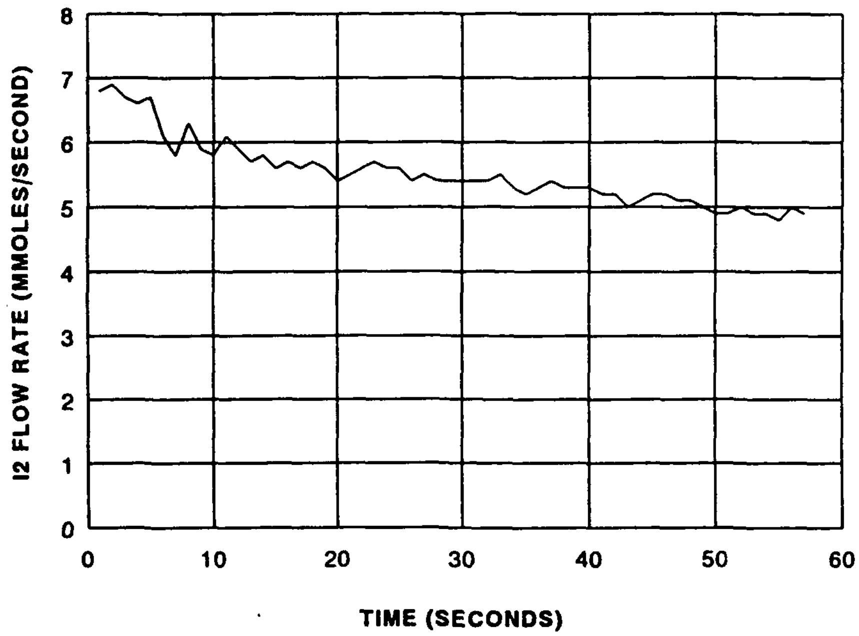

Figure 5. Variation of iodine flow on the RADICL device in the absence of active control.

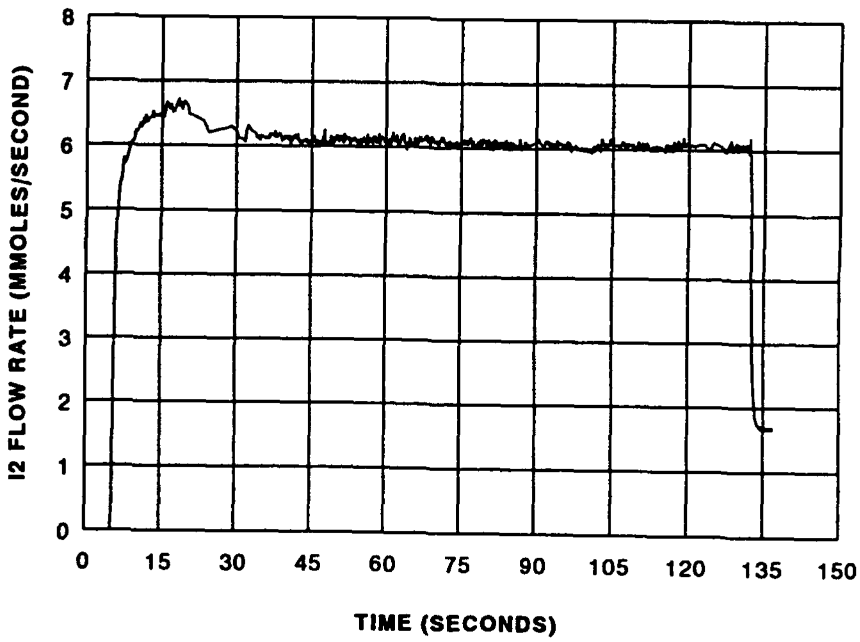

Figure 6. Variation of iodine flow on the RADICL device in the presence of active control.

Some results from low iodine flow tests on a much smaller test stand (IF) are presented in Figures 7 and 8. Figure 7 shows a test in which the initial helium run flow chosen by the user was much lower than required to reach the desired set point of 0.35 mmole/s iodine. Although it took ~80 s to reach the set point, this test illustrates how the control system can obtain the set flow even when given very poor starting parameters. In reality, a test like this would be conducted prior to a lasing test to find good starting parameters for the lasing test. Figure 8 illustrates the same principle in the other direction. The initial helium flow was far too large for the desired iodine flow. Again the helium flow was lowered until the set point was reached.

Although the control system worked well for iodine control on resident COILs, there are modifications that could improve the system. The control computer could keep track of iodine usage over a period of months and warn the operator when more iodine is needed to be added to the boiler. The operating parameters needed for given iodine flows could be stored and cataloged to record a history of iodine system performance. The code could eventually use this database to select starting helium flow rates given the desired iodine flow rate. This would minimize the time required for the iodine flow to stabilize at the desired flow rate during a lasing test. All of these modifications have the common goal of freeing the operator from mundane tasks.

5.0 CONCLUSIONS

An iodine flow rate feedback and control system has been designed that effectively stabilized the iodine flow for two different COIL devices at the Chemical Laser Facility. The system was simple, inexpensive to develop and install, and is adaptable to somewhat larger laser devices without major modification. The development of this system was another step in an ongoing effort to automate the operation of COIL devices.

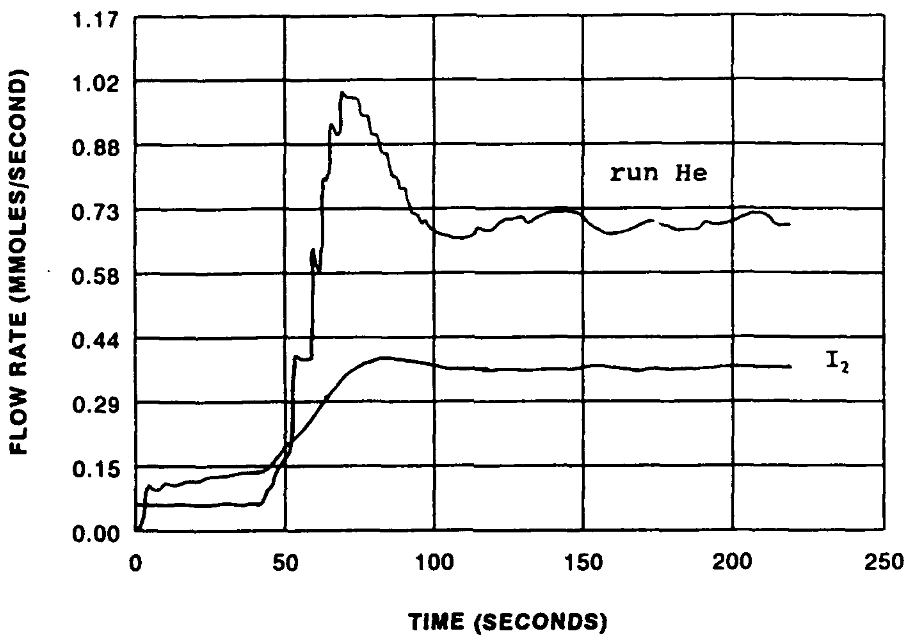

Figure 7. The run helium purposely started low to observe the recovery of the control system.

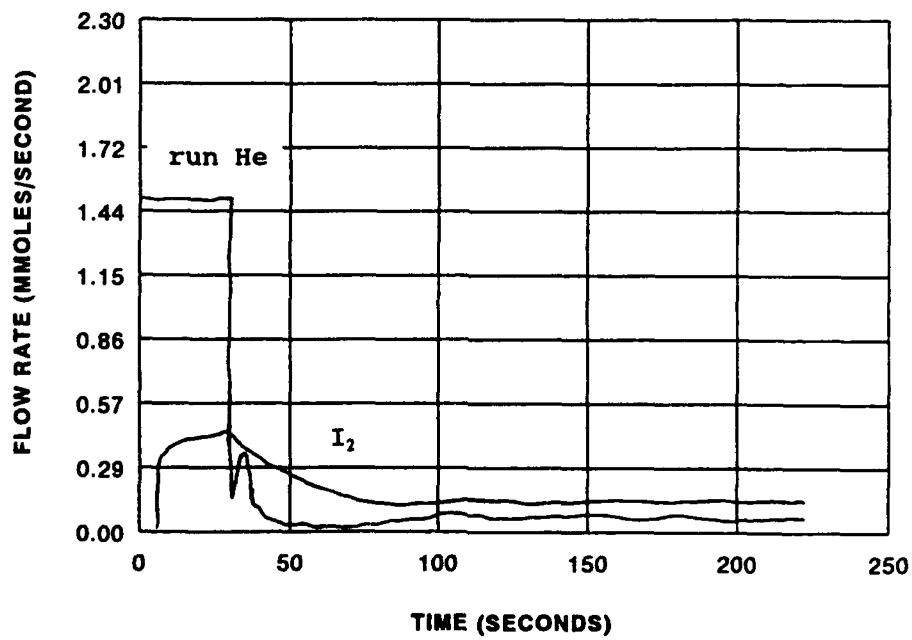

Figure 8. The run helium purposely started high to observe the recovery of control system.

APPENDIX
PROGRAM RADICAL

```
ECHO.OFF
FORGET.ALL
\ **********************   PROGRAM RADICAL.TXT  *****************************
\
\ This program is written in ASYST.  It is used on the RADICAL COIL device
\ to automatically control the flow of gaseous iodine at some user given
\ set point.  The program asks the user for this set point, initial
\ flow for run helium through the boiler, and initial flow for bypass helium.
\ The computer then commands the mass flow controllers to flow at these
\ desired flowrates and asks the user for a keypress to continue.
\ After a keypress, the computer averages a background signal for making
\ an iodine absorption measurement and waits for a second keypress to begin
\ controlling iodine flow.  When this keypress is given, the computer stays
\ in a wait mode in which it reads in data from the diagnostics channels,
\ calculates the iodine flow, and waits until the iodine boiler is opened
\ by the device operator.  When the boiler is opened and the flow has reached
\ a minimum value, the computer waits for the flow to stabilize within 3%.
\ When this point is reached, the computer proportionally modifies the run
\ flow through the boiler, calculates the iodine flow rate, and waits for
\ the flow to stabilize to within 3%. It then compares this flow with the
\ desired iodine flow and continues modifying the flow to keep it as close
\ as possible to the user's desired flow rate. The computer also adjusts
\ the bypass helium flow to maintain a constant total flow through the
\ boiler and the bypass throughout the run.  When the boiler is closed,
\ the computer waits until the iodine flow has reduced to 25% of its peak
\ value and then kicks out of the program.  The user can then display a graph
\ of all iodine flow rates and partial pressures calculated during the run.
\
\  Written by Dr. Charlie Helms and 2Lt Matt Murdough
\  PL/LIDB Chemical Laser Facility
\  Last Modification: 24 FEB 92

\ ******************************************************************************


\ **********************   VARIABLE DECLARATION *****************************

REAL
SCALAR BKGRD.I2.VOLTAGE
SCALAR PI2
SCALAR I2.FLOW
SCALAR STABLE.CHECK.TOTAL
SCALAR STABLE.CHECK.AVERAGE
SCALAR CELL.TEMPERATURE          \ CHANNEL 0    These are the channels
SCALAR CELL.PRESSURE             \ CHANNEL 1     of the A/D board at which
SCALAR RUN.PRESSURE.UP           \ CHANNEL 2     the various diagnostics
SCALAR BYPASS.PRESSURE.UP        \ CHANNEL 3     signals are handled.
SCALAR I2.SIGNAL                 \ CHANNEL 4
SCALAR RUN.FLOW                  \ CHANNEL 5
SCALAR BYPASS.FLOW               \ CHANNEL 6
SCALAR FIRST.CHECK
SCALAR SECOND.CHECK
SCALAR I2.NUMBER.DENSITY
SCALAR IO
SCALAR DESIRED.FLOWRATE
SCALAR I2.FLOW
```

```
SCALAR MAX.I2.FLOW
SCALAR NEW.RUN.FLOW
SCALAR CONVERSION
SCALAR NEW.RUN.DIGITAL
SCALAR TOTAL.MFC.HELIUM.FLOW
SCALAR NEW.BYPASS.FLOW
SCALAR NEW.BYPASS.DIGITAL
SCALAR T.TYPE.MULTIPLIER
SCALAR T.TYPE.OFFSET
SCALAR A0
SCALAR A1
SCALAR A2
SCALAR A3
SCALAR A4
SCALAR A5
SCALAR A6
SCALAR A7
SCALAR X
SCALAR INITIAL.RUN.FLOW
SCALAR INITIAL.BYPASS.FLOW
SCALAR RUN.CONVERSION
SCALAR BYPASS.CONVERSION
SCALAR PERCENT.INCREASE
SCALAR RUN.DIGITAL.INCREASE
SCALAR BYPASS.DIGITAL.INCREASE
SCALAR TOLERANCE
SCALAR RUN.ORIFICE.CONSTANT
SCALAR BYPASS.ORIFICE.CONSTANT
SCALAR RUN.ORIFICE.FLOW
SCALAR BYPASS.ORIFICE.FLOW
SCALAR TOTAL.ORIFICE.FLOW
SCALAR TOTAL.MFC.FLOW
SCALAR TOTAL.NEW.FLOW
SCALAR TOTAL.DIFF.FLOW
SCALAR B
REAL DIM[ 30 ] ARRAY PARTIAL.PRESSURE.ARRAY  \ Stores iodine flow and partial
REAL DIM[ 30 ] ARRAY FLOW.RATE.ARRAY         \ pressures for graphing.

\ ***************** INITIALIZE VARIABLES  ***************************


: INITIALIZE.VARIABLES
    0. TOTAL.MFC.HELIUM.FLOW :-
    0. I2.NUMBER.DENSITY :-
    0. BKGRD.I2.VOLTAGE :-
    0. PI2 :-
    0. I2.FLOW
    0. CELL.TEMPERATURE :-
    0. CELL.PRESSURE :-
    0. I2.SIGNAL :-
    0. RUN.FLOW :-
    0. BYPASS.FLOW :-
    0. MAX.I2.FLOW :-
   9.52 RUN.CONVERSION :-             \ Conversion for going from mmole/sec
   9.52 BYPASS.CONVERSION :-          \ to bits for D/A output.
    0. PERCENT.INCREASE :-
```

```
    0. RUN.DIGITAL.INCREASE :-
    0. BYPASS.DIGITAL.INCREASE :-
    0.03 TOLERANCE :-                       \ 3% for use in STABILITY.CHECK
    0. RUN.PRESSURE.UP :-
    0. BYPASS.PRESSURE.UP :-
    1.39 RUN.ORIFICE.CONSTANT :-             \ Converts pressure to mmole/sec
    8.81 BYPASS.ORIFICE.CONSTANT :-          \ at STP (sonic orifice equation)
    0. RUN.ORIFICE.FLOW :-
    0. BYPASS.ORIFICE.FLOW :-
    0. TOTAL.ORIFICE.FLOW :-
    0. TOTAL.MFC.FLOW :-
    0. TOTAL.NEW.FLOW :-
    0. TOTAL.DIFF.FLOW :-
    0. PARTIAL.PRESSURE.ARRAY :-
    0. B :-
    0. FLOW.RATE.ARRAY :-

        -0.003378      T.TYPE.OFFSET :-              \ These are the values
         0.0024247     T.TYPE.MULTIPLIER :-          \ used in the polynomial
         0.10086091    A0 :-                         \ temperature fit for
     25727.94369       A1 :-                         \ thermocouple voltage to
   -767345.8295        A2 :-                         \ temperature.
  78025595.81          A3 :-
  -9247486589.0        A4 :-
        6.97688E11     A5 :-
       -2.66192E13     A6 :-
        3.94078E14     A7 :-
;

\ *************************** ND--GD *****************************************

: ND                                  \ Allows for quick conversion from
  NORMAL.DISPLAY                      \ normal display mode to graphics
;                                     \ display mode or vice versa.
: GD
  GRAPHICS.DISPLAY
;

\ ****************   SET UP A/D AND D/A TEMPLATES   ********************

0 6 A/D.TEMPLATE READ.ALL.SIGNALS           \ Initializes templates for
4 4 A/D.TEMPLATE READ.I2.CHANNEL            \ ASYST to read from the A/D
0 1 D/A.TEMPLATE WRITE.DATA                 \ board on A/D channels 0-6 and
LOAD.OVERLAY WAVEOPS.SOV                    \ read in on D/A channels 0-1.


\ ***************************  CENTER   *********************************

: CENTER                                   \ Centers the cursor and does a
   SCREEN.CLEAR                            \ screen clear for screen output
   11 0 DO CR LOOP                         \ purposes.
;

\ *********************  GET FLOWRATE  *********************************
\
\ This subroutine asks the user for the desired iodine flow rate to be
```

```
\ used throughout the program.
\
\ ************************************************************************

: GET.FLOWRATE
    CENTER
    ." What iodine flowrate do you desire (mmol/sec)?   "
    #input DESIRED.FLOWRATE :-
;

\ ******************** WRITE INITIAL FLOWS ******************************
\
\ This subroutine asks the user for the initial run flow and bypass flow.
\ The computer then converts these values to their digital equivalents and
\ send them out through the D/A output.  The resultant voltage commands the
\ mass flow controllers to flow at the user requested rates.
\
\ ************************************************************************

: WRITE.INITIAL.FLOWS
   CR CR

   ." What MFC helium run flowrate do you want (0-107 mmoles/sec)? "
   #INPUT INITIAL.RUN.FLOW :-
   CR CR ." What MFC helium bypass flowrate do you want (0-107 mmoles/s)? "
   #INPUT INITIAL.BYPASS.FLOW :-
   INITIAL.RUN.FLOW INITIAL.BYPASS.FLOW + TOTAL.MFC.HELIUM.FLOW :-
   INITIAL.RUN.FLOW RUN.CONVERSION * 2048 + INITIAL.RUN.FLOW :-
   INITIAL.BYPASS.FLOW BYPASS.CONVERSION * 2048 + INITIAL.BYPASS.FLOW :-
   WRITE.DATA
   D/A.INIT
   INITIAL.BYPASS.FLOW INITIAL.RUN.FLOW D/A.OUT
;


\ ********************    CALCULATE BACKGROUND    **********************
\
\ Collects voltage signals from the iodine signal lock-in for 10 seconds
\ and averages the collected data.  The final value is used to in
\ CALC.I2.FLOW to calculate the flow of iodine.
\
\ *********************************************************************

: CALCULATE.BACKGROUND
    READ.I2.CHANNEL
    A/D.INIT
    0. IO :-
    100 0 DO          \ Collect background voltage for 10 seconds at 10 Hz
        A/D.IN
        IO + IO :-
        100. MSEC.DELAY
    LOOP
    IO 100 / 409.5 /  BKGRD.I2.VOLTAGE :-
    ."  The background iodine signal (IO) is " BKGRD.I2.VOLTAGE . ." volts"
;
```

```
\ ************************* READ DATA  *********************************
\
\ This subroutine reads the data from channels 0-6 that the computer
\ uses to calculate iodine flow rate. It then converts the digital
\ information to volts.
\
\ ***************************************************************************

: READ.DATA

  READ.ALL.SIGNALS
  A/D.INIT
  A/D.IN
  409.5 / BYPASS.FLOW :-
  409.5 / RUN.FLOW :-                    \ 409.5 is the conversion from
  409.5 / I2.SIGNAL :-                   \ bits to volts
  409.5 / BYPASS.PRESSURE.UP :-
  409.5 / RUN.PRESSURE.UP :-
  409.5 / CELL.PRESSURE :-
  409.5 / CELL.TEMPERATURE :-
;

\ ********************  CONVERT MEDIA DIAGNOSTICS  ************************
\
\ This subroutine converts the data obtained in READ.DATA from volts to
\ conventional units.  It performs the polynomial thermocouple fit to
\ obtain temperatures and uses a simplified sonic orifice calculation to
\ find bypass orifice helium flow rate.  If the calculated cell pressure
\ is greater than 2000 Torr, the computer shuts off the mass flow controller
\ flows and warns the user with a bell that the cell has over-pressurized.
\
\ ***************************************************************************

:  CONVERT.MEDIA.DIAGNOSTICS

   CELL.TEMPERATURE T.TYPE.MULTIPLIER * T.TYPE.OFFSET + X :-
   X A7 * A6 + X * A5 + X * A4 +
   X * A3 + X * A2 + X * A1 + X * A0 + CELL.TEMPERATURE :-

   RUN.PRESSURE.UP 50 * RUN.PRESSURE.UP :-          \ in psi
   BYPASS.PRESSURE.UP 50 * BYPASS.PRESSURE.UP :-    \ in psi
   CELL.PRESSURE  100 * CELL.PRESSURE :-            \ in Torr 1000 torr head
   RUN.FLOW 21.5 * RUN.FLOW :-                      \ in mmoles/sec
   BYPASS.FLOW 21.5 * BYPASS.FLOW :-                \ in mmoles/sec
   RUN.FLOW BYPASS.FLOW + TOTAL.MFC.FLOW :-         \ in mmoles/sec

   RUN.PRESSURE.UP RUN.ORIFICE.CONSTANT * RUN.ORIFICE.FLOW :-
   BYPASS.PRESSURE.UP BYPASS.ORIFICE.CONSTANT * BYPASS.ORIFICE.FLOW :-
   RUN.ORIFICE.FLOW BYPASS.ORIFICE.FLOW + TOTAL.ORIFICE.FLOW :-


\ ********* SHUT SYSTEM DOWN IF IODINE CELL OVER PRESSURIZES ***************

   CELL.PRESSURE 2000. >
```

```
        IF
            WRITE.DATA
            D/A.INIT
            0 0 D/A.OUT
            CENTER
            ." W A R N I N G !!!     CELL PRESSURE HAS EXCEEDED 2000 TORR"
            CR CR
            ."     FLOWS HAVE BEEN SHUT OFF"
            10 0 DO BELL LOOP
            QUIT
        THEN

\ ***************************************************************************

;

\ **********************  STABLE CHECK LOOP  *******************************
\
\ This subroutine reads the iodine signal channel and averages 10 samples
\ at a 10 Hz rate. This average is used to check for flow stability by
\ the STABILITY.CHECK subroutine.
\
\ ***************************************************************************

: STABLE.CHECK.LOOP

      0. STABLE.CHECK.TOTAL  :=
      0. STABLE.CHECK.AVERAGE :=
      10 0 DO
          A/D.IN
          STABLE.CHECK.TOTAL + STABLE.CHECK.TOTAL :=
      LOOP
      STABLE.CHECK.TOTAL 10 / STABLE.CHECK.AVERAGE :=
;

\ ************************  STABILITY CHECK  *******************************
\
\ This subroutine takes an average from STABLE.CHECK.LOOP, waits 3 seconds,
\ and takes another average.  It then compares the two averages to see if
\ they are within 3% (variable TOLERANCE) of each other.  If they are, then
\ the computer kicks out of the subroutine.  If they are not within 3% of
\ each other, i.e. the iodine flow is not stable, then the subroutine repeats
\ another 3 second averaging and comparison and continues to do so until
\ the flow is stabilized.
\
\ ***************************************************************************

: STABILITY.CHECK

    READ.I2.CHANNEL
    A/D.INIT
    BEGIN
      STABLE.CHECK.LOOP
      STABLE.CHECK.AVERAGE FIRST.CHECK :=
      3000. MSEC.DELAY
      STABLE.CHECK.LOOP
```

```
    STABLE.CHECK.AVERAGE SECOND.CHECK :=
    FIRST.CHECK SECOND.CHECK -
    SECOND.CHECK / ABS  TOLERANCE <
  UNTIL
  CR CR ." IODINE FLOW STABILIZED AT " SECOND.CHECK 409.5 / . ." VOLTS"
;


\ ************************  CALC I2 FLOW  *********************************
\
\ In this subroutine, the computer performs a Beer's Law calculation to
\ determine the iodine partial pressure and flow rate. It then stores the
\ values calculated each time this subroutine is called into arrays for
\ iodine partial pressures and flow rates for use in graphs after the run
\ is finished. MAX.I2.FLOW is a running value of the highest flow rate
\ calculated during a given run.  It is used in the main RUN routine
\ to determine when to kick out of the program.
\
\ *************************************************************************

: CALC.I2.FLOW

  I2.SIGNAL BKGRD.I2.VOLTAGE /                      \ Beer's Law:
  LN NEG 1.7E-18 / 2.568 /  I2.NUMBER.DENSITY :=    \ 1.7E-18 = cross section
                                                    \ 2.568 = path length


    I2.NUMBER.DENSITY  .0821 * 760 * 1000 *        \ Ideal Gas Law:
    CELL.TEMPERATURE 273 + * 6E23 /  PI2 :=        \ .0821 = R, 760 Torr= latm

                                                   \ 273 K = 0 Celsius,
                                                   \ 1000 mmole=1 mole
                                                   \ 6E23 = Avagadro's Number

    PI2 CELL.PRESSURE / TOTAL.MFC.FLOW TOTAL.ORIFICE.FLOW + * I2.FLOW :=

    MAX.I2.FLOW I2.FLOW <
    IF       I2.FLOW MAX.I2.FLOW :=    THEN
    CR CR ." CALCULATED I2 FLOW IS  " I2.FLOW .

    B 1 + B :=
    PI2 PARTIAL.PRESSURE.ARRAY [ B ] :=
    I2.FLOW FLOW.RATE.ARRAY [ B ] :=

;

\ ***********************  MODIFY FLOWS  **********************************
\
\ This subroutine modifies the flow of helium through the iodine boiler
\ and through the bypass.  It compares the current measured iodine flow rate
\ with the desired flow rate and makes the new run helium flow equal to
\ 0.8 times 1 minus the fraction of desired flow over measured flow. Thus the
\ iodine flow rate is forced to proportionally approach the user's desired
\ flow rate. With each change in the flow rate, the calculated iodine
\ flow and the change in helium flows is printed to the screen. The
\ computer also changes the bypass helium flow so that the total
```

```
\ run and bypass flow is constant at whatever total the user entered
\ in WRITE.INITIAL.FLOWS.
\
\ ***************************************************************************

: MODIFY.FLOWS
    0. RUN.ORIFICE.FLOW :=
    DESIRED.FLOWRATE I2.FLOW / 1 - .8 * 1 + RUN.ORIFICE.FLOW RUN.FLOW + *
    TOTAL.NEW.FLOW :=
    TOTAL.NEW.FLOW RUN.ORIFICE.FLOW - NEW.RUN.FLOW :=


    NEW.RUN.FLOW 107.0 >
    IF
      107.0 NEW.RUN.FLOW :=
      CR CR ."       RUN HELIUM FLOW IS MAXED OUT "    \ Tells user the flow
    THEN                                               \ controller is at
                                                       \ 100% of capacity
    NEW.RUN.FLOW .05 <
    IF                            \ Prevents divide by zero error
      .05 NEW.RUN.FLOW :=
    THEN
    NEW.BYPASS.FLOW .05 <         \ Prevents divide by zero error
    IF
      .05 NEW.BYPASS.FLOW :=
    THEN

    NEW.RUN.FLOW RUN.CONVERSION * 2048 + NEW.RUN.DIGITAL :=
    TOTAL.MFC.HELIUM.FLOW NEW.RUN.FLOW - NEW.BYPASS.FLOW :=
    NEW.BYPASS.FLOW BYPASS.CONVERSION * 2048 + NEW.BYPASS.DIGITAL :=
    WRITE.DATA
    D/A.INIT
    NEW.BYPASS.DIGITAL NEW.RUN.DIGITAL  D/A.OUT

    NEW.RUN.FLOW RUN.FLOW / 1 >
    IF
      CR CR ." RUN HELIUM FLOW INCREASED BY " NEW.RUN.FLOW RUN.FLOW / 1 -
      100 * . ." %"
    THEN
    NEW.RUN.FLOW RUN.FLOW / 1 <
    IF
      CR CR ." RUN HELIUM FLOW DECREASED BY " NEW.RUN.FLOW RUN.FLOW / 1 -
      -100 * . ." %"
    THEN

;

\ **************************  GPP and GI2 ********************************
\
\ These two subroutines make plots of the iodine partial pressure and
\ flow rate calculations made and stored throughout the run.
\
\ ************************************************************************
: GPP
```

```
PARTIAL.PRESSURE.ARRAY  Y.AUTO.PLOT

;

: GI2

FLOW.RATE.ARRAY  Y.AUTO.PLOT

;

\ ********************************  RUN  **********************************
\
\ Run is the main program routine.  It calls the various subroutines
\ throughout the program.  It initializes the variables, asks the user
\ for the initial flows and waits for a keypress to collect the background.
\ After collecting the background, it waits for another keypress to begin
\ modifying flows.  When this keypress is received, it checks to see if
\ the iodine signal is stable; if it is, then it reads the diagnostic
\ channels and calculates the iodine flow rate.  If the flow rate is less
\ than 0.5 mmoles/sec, it loops through the another stability check and
\ iodine flow rate calculation.  If the flow is greater than 1.0 mmoles/sec
\ it begins to modify flows. After each modification, the computer checks
\ to see if the flow rate has met two conditions: (1) the max iodine seen
\ thus far is greater than 0.9 mmoles\sec, i.e. the boiler has been opened
\ iodine has flowed, and (2) the iodine has dropped to less than 25% of its
\ highest value reached during the run.  If both these conditions are true,
\ then the boiler has been opened and closed by the operator and the run is
\ finished allowing the program to shut off.  If both of these conditions
\ are not met, then the boiler has not been opened or is still open and the
\ the computer remains in a modify flows loop.
\
\ ***************************************************************************

: RUN
    STACK.CLEAR
    INITIALIZE.VARIABLES
    GET.FLOWRATE

    WRITE.INITIAL.FLOWS
    CENTER
    ."              PRESS ANY KEY TO COLLECT BACKGROUND I2 SIGNAL " PCKEY
    CENTER
    CALCULATE.BACKGROUND
    CR CR ."                 PRESS ANY KEY TO BEGIN CONTROLLING FLOWS"  PCKEY
    CENTER
    BEGIN
       STABILITY.CHECK
       READ.DATA
       CONVERT.MEDIA.DIAGNOSTICS
       CALC.I2.FLOW
       I2.FLOW 0.5 >  IF   MODIFY.FLOWS     THEN
       MAX.I2.FLOW 0.9 >
       IF
           I2.FLOW 4 * ABS MAX.I2.FLOW <
           ELSE  FALSE
       THEN
```

```
    UNTIL
    CENTER
    ."              IODINE FLOW IS COMPLETE"
    CR CR   ." TYPE 'GPP' TO PLOT IODINE PARTIAL PRESSURE"
    CR CR   ." TYPE 'GI2' TO PLOT IODINE FLOWRATE"
    CR CR CR

;
```